



(12) **DEMANDE DE BREVET EUROPEEN**

(43) Date de publication:
24.08.2005 Bulletin 2005/34

(51) Int Cl.7: **G06F 1/00, G06F 9/00**

(21) Numéro de dépôt: **05290303.6**

(22) Date de dépôt: **10.02.2005**

(84) Etats contractants désignés:
AT BE BG CH CY CZ DE DK EE ES FI FR GB GR
HU IE IS IT LI LT LU MC NL PL PT RO SE SI SK TR
Etats d'extension désignés:
AL BA HR LV MK YU

(72) Inventeur: **Helou, Didier**
78210 Saint Cyr l'Ecole (FR)

(74) Mandataire: **Chaffraix, Sylvain et al**
COMPAGNIE FINANCIERE ALCATEL
Département Propriété Industrielle
54, rue La Boétie
75411 Paris Cedex 08 (FR)

(30) Priorité: **18.02.2004 FR 0450299**

(71) Demandeur: **ALCATEL**
75008 Paris (FR)

(54) **Procédé et dispositif de transformation d'un système d'exploitation contre des intrusions non autorisées**

(57) Un dispositif (D) est dédié à la transformation d'un système d'exploitation (OS), éventuellement au sein d'un équipement (PC). Ce dispositif (D) comprend

des moyens de traitement (MT) chargés de brouiller l'un au moins des outil(s)-support(s) de programmation (Fst, Fct) du système d'exploitation (OS) par insertion dans sa définition d'au moins un paramètre de brouillage.

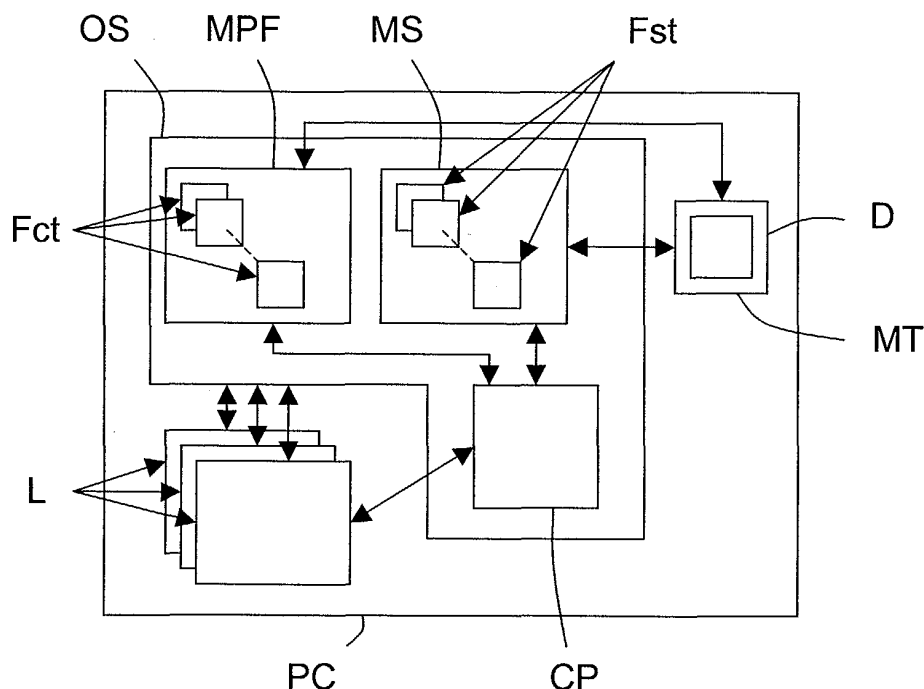


Figure unique

Description

[0001] L'invention concerne le domaine des équipements gérés par un système d'exploitation et un ou plusieurs programmes informatiques compilés avec ce système d'exploitation.

[0002] On entend ici par « équipement » tout type de matériel contrôlable au moyen de programmes compilés, et notamment les ordinateurs, fixes ou portables, les stations de travail, les équipements de réseau, tels que les serveurs ou les routeurs, et les terminaux de communication, fixes ou mobiles, y compris ceux de type multimédia, comme par exemple les téléphones et les assistants personnels numériques (ou PDA).

[0003] La quasi totalité des programmes informatiques, voire même des logiciels, compilés à l'aide d'un système d'exploitation (ou OS pour « Operating System »), quel qu'il soit, peut faire l'objet d'une intrusion externe au moyen d'un ou plusieurs programmes, généralement appelés « binaires » du fait qu'ils sont constitués de codes binaires.

[0004] Ces binaires sont fréquemment classés au sein de familles, telles que les vers, les virus, les chevaux de Troie ou les espions ou mouchards, selon leur mode d'action. Ils permettent à leurs concepteurs de récupérer au sein d'un équipement des informations stockées ou saisies en temps réel, ou de contrôler un équipement à distance, ou de détruire des données ou des morceaux de programmes ou logiciels internes stockés dans un équipement (y compris ceux constituant l'OS), ou encore de contraindre des programmes ou logiciels internes, stockés dans un équipement, à exécuter des codes binaires non autorisés, par exemple pour les contraindre à effectuer des démonstrations ou les soumettre à des tests.

[0005] Ces binaires profitent de défauts de sécurité que présentent les programmes internes. Il est certes possible de remédier à ces défauts au moyen de programmes (ou « patches ») correctifs, cependant, le temps nécessaire au développement d'un programme correctif, dédié à un binaire, et à l'exécution de tests dits de régression est rarement inférieur à une semaine, ce qui permet au binaire de continuer d'opérer, et peut permettre à d'autres binaires d'opérer librement. En outre, l'ajout d'un programme correctif peut parfois perturber, voire empêcher, le fonctionnement d'applications.

[0006] Par ailleurs, certains programmes ou logiciels internes peuvent créer de façon non intentionnelle des portes d'accès (ou « security hole ») pour des binaires externes, ou bien supprimer des portes de sécurité initialement destinées à bloquer des binaires externes, dans des programmes ou logiciels internes avec lesquels ils coopèrent. Cela peut rendre impossible la restauration de certains fichiers d'installation, et par conséquent imposer une restauration complète d'une installation informatique.

[0007] L'invention a donc pour but d'améliorer la situation.

[0008] Elle propose à cet effet un procédé de protection d'un programme informatique interne contre des intrusions extérieures effectuées au moyen d'au moins un programme informatique externe. Ce programme informatique interne fonctionne sur un équipement muni d'un système d'exploitation. Le procédé se caractérise en ce qu'il consiste à brouiller le système d'exploitation puis à compiler le programme interne avec le système d'exploitation brouillé.

[0009] Pour brouiller le système d'exploitation, préférentiellement on le transforme par un brouillage de l'un au moins de ses outil(s)-support(s) de programmation consistant en une insertion dans sa définition d'au moins un paramètre de brouillage.

[0010] De préférence, on brouille tous les outil(s)-support(s) de programmation. En outre, lorsque chaque outil-support est défini par un multiplet comportant des paramètres ou des variables (associés à un type), il est également préférable d'insérer un paramètre de brouillage avant ou après chaque paramètre ou variable d'un multiplet.

[0011] Le brouillage peut également consister à permuter au moins deux variables ou paramètres, et de préférence toutes, de l'un au moins desdits multiplets, et de préférence tous, en complément de l'insertion de paramètre(s) de brouillage.

[0012] Le brouillage peut être effectué en fonction d'une loi choisie, de préférence variable, par exemple de façon pseudo-aléatoire.

[0013] Les outil(s)-support(s) de programmation sont préférentiellement choisis parmi des prototypes de fonction (ou « fonction prototypes ») et des fichiers inclus internes (ou « internal include files ») définissant chacun une structure.

[0014] Par exemple, chaque prototype de fonction brouillé (Fct) peut être défini par un multiplet se présentant sous la forme « Fct(typei Pi,[type Dummyi,] typej Pj,[type Dummyj,]... typer Pr,[type Dummyr]) », où « typei » est le type i du paramètre d'appel Pi de la fonction concernée, et « Dummyi » représente un ou plusieurs paramètres de brouillage insérés, associés au paramètre d'appel Pi. De même, chaque fichier inclus interne brouillé (Fst) peut être défini par un multiplet se présentant sous la forme Fst{typei Di,[type Dummyi,] typej Dj,[type Dummyj,]... typer Dr,[type Dummyr]}, où « typei » est le type i de la variable Di de la structure concernée, et « Dummyi » représente un ou plusieurs paramètres de brouillage insérés, associés à la variable Di.

[0015] L'invention propose également un dispositif informatique dédié à protection d'un programme informatique et comprenant des moyens de traitement chargés de mettre en oeuvre un procédé du type de celui présenté ci-avant.

[0016] L'invention est particulièrement bien adaptée, bien que de façon non exclusive, au brouillage de systèmes d'exploitation tels que ceux connus sous les noms « LINUX », « BSD », « Solaris », « Tru64 » et « WINDOWS » (mar-

ques déposées).

[0017] D'autres caractéristiques et avantages de l'invention apparaîtront à l'examen de la description détaillée ci-après, et du dessin annexé, sur lequel l'unique figure illustre de façon très schématique un exemple d'ordinateur équipé d'un dispositif de transformation selon l'invention. Le dessin annexé pourra non seulement servir à compléter l'invention, mais aussi contribuer à sa définition, le cas échéant.

[0018] L'invention a pour objet de permettre la protection de programmes informatiques (ou logiciels) contre des intrusions externes effectuées à l'aide d'autres programmes informatiques, comme par exemple des binaires.

[0019] Cette invention concerne tout type de programme informatique dès lors que le programme doit faire l'objet d'une compilation à l'aide d'un compilateur, faisant partie d'un système d'exploitation ou OS (pour « Operating System »), pour fonctionner au sein d'un équipement. Par ailleurs, l'invention concerne tout type de système d'exploitation, qu'il soit de type monotâche ou multitâches, et notamment ceux appelés « UNIX », « LINUX », « BSD », « Solaris », « Tru64 », « OS/2 », « BeOS », « MS-DOS », « WINDOWS » et « OS MAC » (marques déposées).

[0020] On considère dans ce qui suit que le système d'exploitation est LINUX et qu'il est implanté dans un ordinateur, fixe ou portable. Bien entendu, il pourrait être implanté, ou destiné à être implanté, dans tout autre type d'équipement contrôlé au moins partiellement par des programmes informatiques compilés, et notamment dans une station de travail ou un équipement de réseau, tel qu'un serveur ou un routeur, ou bien dans un terminal de communication, fixe ou mobile, éventuellement de type multimédia, tel qu'un téléphone ou un assistant personnel numérique (ou PDA).

[0021] Comme cela est illustré sur l'unique figure, un ordinateur PC comporte habituellement un système d'exploitation OS couplé à un ou plusieurs programmes informatiques ou logiciels applicatifs L dédiés, par exemple, à la messagerie (pour l'émission et la réception de courriers électroniques), ou à l'accès à un réseau (privé ou public, tel qu'Internet), ou au traitement de texte ou de photos, ou à la lecture et/ou l'enregistrement de données numériques, ou à des jeux, ou encore à la simulation numérique.

[0022] Un système d'exploitation OS est un logiciel chargé de contrôler le fonctionnement d'un ordinateur PC, et notamment de gérer l'allocation et l'utilisation de ressources matérielles, comme par exemple la mémoire, l'unité centrale de traitement (ou CPU), l'espace du disque dur et les périphériques. Il sert en outre d'interface de commande avec l'ordinateur PC et notamment avec les logiciels applicatifs L qu'il contient.

[0023] De façon très simplifiée, un système d'exploitation OS comprend au moins un premier module MPF, comportant un premier type d'outil-support de programmation Fct, un second module MS, comportant un second type d'outil-support de programmation Fst, et un compilateur CP chargé de compiler des programmes ou logiciels internes L à l'aide des outils-soutiens des premier MPF et second MS modules afin qu'ils puissent fonctionner au sein de l'ordinateur PC.

[0024] On entend ici par « outil-support du premier type » ce que l'homme de l'art appelle habituellement un prototype de fonction Fct. Un tel prototype de fonction Fct constitue une description sémantique d'une fonction de programmation et définit fréquemment une interface de programmation d'application (ou API pour « Application Programming Interface »). Il est habituellement défini par un multiplét désigné par un nom et se présentant sous la forme Fct(type1 P1, type2 P2,..., typen Pn), où « typei » est le type i (i = 1 à n, où n varie selon la fonction) d'un paramètre d'appel Pi de la fonction concernée. Il existe généralement huit types différents : quatre scalaires (booléen, entier, nombre à virgule flottante, et chaîne de caractères), deux composés (tableau et objet), et deux spéciaux (ressource et Null). On peut également prévoir des types mixtes pour les paramètres d'appel Pi pouvant posséder des types différents.

[0025] On peut citer à titre d'exemple les fonctions Noyau (ou « Kernel »), Bibliothèque (ou « Library »), Pilote (ou « Driver »), et Application. Un exemple illustratif de fonction Kernel est donné ci-dessous, en langue anglaise, avec des commentaires en français :

```
static ssize_t module_output(
    struct file *file,          /* Le fichier lu */
    char *buf,                 /* La mémoire tampon à utiliser */
    size_t len,                /* La longueur de la mémoire tampon */
    off_t *offset)             /* Décalage dans le fichier - ignore */
```

[0026] Par ailleurs, on entend ici par « outil-support du second type » ce que l'homme de l'art appelle habituellement un fichier inclus interne (ou « internal include file ») Fst. Un tel fichier inclus interne Fst constitue une description sémantique d'une structure de programmation utilisée dans un programme. Il est habituellement défini par un multiplét

désigné par un nom et se présentant sous la forme Fst{type1 D1; type2 D2;... ; typem Dm}, où « typei » est le type i (i = 1 à m, où m varie selon la structure) d'une variable Di de la structure concernée. Les types sont généralement les mêmes que ceux utilisés pour les prototypes de fonction Fct.

[0027] Généralement, on appelle une fonction avec une structure associée. Un exemple illustratif de structure faisant appel à plusieurs variables Di est donné ci-dessous, en langue anglaise, avec des commentaires en français :

```

struct zatm_vcc {
    int rx_chan;      /* canal RX, 0 si aucun */
    int pool;         /* ensemble de mémoires tampon */
    int tx_chan;      /* canal TX, 0 si aucun */
    int shaper;       /* profileur, < 0 si aucun */
    struct sk_buff_head tx_queue; /* liste de mémoires tampon */
    wait_queue_head_t tx_wait;  /* pour fermer */
    u32 *ring;        /* anneau de transmission */
    int ring_curr;     /* position courante d'écriture */
    int txing;         /* nombre de transmis */
    struct sk_buff_head backlog; /* liste de mémoires tampon */
};

```

[0028] Afin de protéger les programmes informatiques (ou logiciels) contre des intrusions externes effectuées à l'aide d'autres programmes informatiques, comme par exemple des binaires, l'invention propose de les compiler avec un compilateur CP traditionnel, une fois que le système d'exploitation OS a été brouillé (ou « scrambled »).

[0029] Le brouillage du système d'exploitation OS (par exemple celui appelé « LINUX » (marque déposée)), selon l'invention, consiste au minimum à insérer au moins un paramètre de brouillage factice dans la définition de l'un au moins de ses outil(s)-support(s) de programmation appartenant aux premier Fct et second Fst types.

[0030] L'insertion a pour objet de décaler l'ordre des paramètres dans la pile (ou « stack ») où ils sont stockés. Ainsi, un binaire externe, qui n'a pas été compilé avec le même système d'exploitation OS que le logiciel interne qu'il « attaque », va recevoir des codes d'erreur en réponse à ses requêtes ou va être inutilisable très rapidement (entraînant ainsi un « crash » de l'application).

[0031] La protection conférée par le brouillage, à un logiciel (ou programme) compilé à l'aide d'un système d'exploitation brouillé, est cependant d'autant plus efficace que le nombre d'outils-supports brouillés est élevé. En d'autres termes, il est préférable de brouiller tous les outils-supports des premier et second types, c'est-à-dire tous les prototypes de fonction Fct et tous les fichiers inclus internes Fst.

[0032] Le brouillage peut consister à insérer un ou plusieurs paramètres de brouillage « Dummy » avant ou après l'un au moins des paramètres d'appel Pi ou l'une au moins des variables Di. Un paramètre de brouillage Dummy peut être sélectionné en fonction d'une loi, qui peut varier d'un système d'exploitation à l'autre, et éventuellement d'un outil-support Fct du premier type à un outil-support Fst du second type. Préférentiellement, la loi varie de façon pseudo-aléatoire.

[0033] Par ailleurs, un paramètre de brouillage Dummy peut être de type variable. Il peut être, par exemple, constitué d'un ou plusieurs octets, voire même d'une suite d'octets.

[0034] En outre, la protection est encore plus efficace lorsque l'on insère un paramètre de brouillage devant ou derrière chaque paramètre Pi ou variable Di d'un multiplet définissant une fonction ou une structure. Un tel brouillage aboutit alors aux définitions brouillées suivantes, respectivement pour chaque prototype de fonction Fct et chaque fichier inclus interne Fst, et lorsqu'il est appliqué après les paramètres d'appel Pi ou les variables Di :

- Fct(type1 P1, [type Dummy1,] type2 P2,[type Dummy2,..., typen Pn, [type Dummyn])
- Fst{type1 D1; [type Dummy1;] type2 D2; [type Dummy2;]... ; typem Dm; [type Dummym]}

[0035] En variante, lorsque le brouillage est appliqué avant les paramètres d'appel P_i ou les variables D_i , on obtient les définitions brouillées suivantes :

- $Fct([type\ Dummy1,] type1\ P1, [type\ Dummy2,] type2\ P2, \dots, [type\ Dummyn,] typen\ Pn)$
- $Fst([type\ Dummy1,] type1\ D1; [type\ Dummy2,] type2\ D2; \dots; [type\ Dummym,] typem\ Dm)$

[0036] Dans cet exemple de brouillage, les différents paramètres de brouillage $Dummy_i$ peuvent être sélectionnés en fonction d'une loi, qui peut varier d'un système d'exploitation à l'autre, et éventuellement d'un outil-support Fct du premier type à un outil-support Fst du second type. Préférentiellement, la loi varie de façon pseudo-aléatoire.

[0037] La protection peut être encore plus efficace si le brouillage consiste non seulement en une insertion d'un ou plusieurs paramètres de brouillage mais également en une permutation d'au moins deux paramètres d'appel P_i , P_j ou variables D_i , D_j au sein d'un ou plusieurs multiplets de définition, et de préférence dans chaque multiplet de définition.

[0038] L'efficacité de protection optimale est obtenue lorsque l'on permute tous les paramètres d'appel P_i et toutes les variables D_i au sein de chaque multiplet de définition de fonction et de structure. Un tel brouillage aboutit alors aux définitions brouillées suivantes, respectivement pour chaque prototype de fonction Fct et chaque fichier inclus interne Fst , et lorsqu'il est appliqué après les paramètres d'appel P_i ou les variables D_i :

- $Fct(type_i\ P_i, [type\ Dummy_i,] type_j\ P_j, [type\ Dummy_j,] \dots, type_r\ P_r, [type\ Dummy_r])$
- $Fst([type_j\ D_j; [type\ Dummy_j,] type_r\ D_r; [type\ Dummy_r,] \dots; type_i\ D_i; [type\ Dummy_i])$

[0039] En variante, lorsque le brouillage est appliqué avant les paramètres d'appel P_i ou les variables D_i , on obtient les définitions brouillées suivantes :

- $Fct([type\ Dummy_i,] type_i\ P_i, [type\ Dummy_j,] type_j\ P_j, \dots, [type\ Dummy_r,] type_r\ P_r)$
- $Fst([type\ Dummy_j,] type_j\ D_j; [type\ Dummy_r,] type_r\ D_r; \dots; [type\ Dummy_i,] type_i\ D_i)$

[0040] Dans cet exemple de brouillage, les permutations, ainsi qu'éventuellement les différents paramètres de brouillage $Dummy_i$, peuvent être sélectionnés en fonction d'une loi, qui peut varier d'un système d'exploitation à l'autre, et éventuellement d'un outil-support Fct du premier type à un outil-support Fst du second type. Préférentiellement, la loi varie de façon pseudo-aléatoire.

[0041] Une fois qu'un système d'exploitation OS a été brouillé, conformément à l'invention, il peut être utilisé pour protéger un programme informatique contre des intrusions extérieures. Pour ce faire, il suffit en effet de compiler le programme informatique avec le compilateur CP du système d'exploitation OS brouillé, puisque ce compilateur CP va utiliser ses outils-supports brouillés.

[0042] Comme indiqué précédemment, un binaire externe, qui n'a pas été compilé avec le même système d'exploitation OS que le logiciel qu'il « attaque », va recevoir des codes d'erreur en réponse à ses requêtes ou va être inutilisable très rapidement. Certes, un binaire qui ne ferait pas appel aux APIs du système d'exploitation brouillé pourrait s'introduire dans un programme protégé selon l'invention, mais ses actions seraient alors très réduites, notamment le rapatriement de données via internet serait très difficile à mettre en oeuvre puisque les fonctions standard TCP/IP seraient inutilisables.

[0043] On peut mettre en oeuvre le brouillage à l'aide d'un dispositif de protection D, selon l'invention. Un tel dispositif D n'a besoin que d'un module de traitement MT chargé de brouiller un ou plusieurs outils-supports, par insertion de paramètre(s) de brouillage et éventuellement permutation(s) de paramètres d'appel ou de variables. Pour ce faire, le module de traitement MT doit disposer d'un ensemble de paramètres de brouillage, stockés dans une mémoire dédiée (par exemple sous la forme d'une table ou de fichier(s)), et éventuellement d'une loi, comme décrit précédemment, selon sa configuration.

[0044] Ce dispositif de traitement D, et notamment son module de traitement MT, peuvent être réalisés sous la forme de circuits électroniques, de modules logiciels (ou informatiques), ou d'une combinaison de circuits et de logiciels.

[0045] Un tel dispositif D peut soit être intégré à l'intérieur d'un équipement, comme illustré sur l'unique figure, soit se présenter sous la forme d'un boîtier externe, de type périphérique, que l'on raccorde à un équipement, soit encore se présenter sous la forme d'un logiciel de transformation stocké sur un support de mémorisation, comme par exemple un cédérom, un disque magnéto-optique, ou tout autre type de mémoire amovible. Mais, il peut être également implanté dans un accessoire dédié exclusivement à la transformation par brouillage de systèmes d'exploitation, indépendant des équipements devant être équipés desdits systèmes d'exploitation brouillés. Un tel accessoire peut être également agencé de manière à permettre la compilation de logiciels (ou programmes) destinés à fonctionner avec un système d'exploitation qu'il a précédemment brouillé.

[0046] Grâce à l'invention, les logiciels compilés avec un système d'exploitation brouillé sont protégés contre les intrusions reposant sur des appels aux APIs de ce système d'exploitation.

[0047] Par ailleurs, seul un logiciel interne ayant été compilé avec un système d'exploitation brouillé peut désormais utiliser un autre logiciel interne compilé avec ce même système d'exploitation brouillé.

[0048] En outre, l'invention permet de s'affranchir des programmes (ou « patches ») correctifs et des tests de régression associés. Cela permet de réduire les coûts de développement et d'installation, de s'affranchir des adaptations indispensables en cas de modification d'un logiciel, et de ne pas laisser les logiciels en proie aux intrusions pendant les phases de développement et de tests des programmes correctifs.

[0049] De plus, l'invention permet de protéger, contre les intrusions, des logiciels comportant d'origine des défauts de sécurité.

[0050] L'invention ne se limite pas aux modes de réalisation de dispositif de traitement, de procédé de transformation et de procédé de protection décrits ci-avant, seulement à titre d'exemple, mais elle englobe toutes les variantes que pourra envisager l'homme de l'art dans le cadre des revendications ci-après.

Revendications

1. Procédé de protection d'un programme informatique interne (L) contre des intrusions extérieures effectuées au moyen d'au moins un programme informatique externe, ledit programme informatique interne fonctionnant sur un équipement muni d'un système d'exploitation (OS), **caractérisé en ce qu'il** consiste à brouiller ledit système d'exploitation (OS) puis à compiler ledit programme interne (L) avec le système d'exploitation (OS) brouillé.
2. Procédé de protection selon la revendication 1, dans lequel, ledit système d'exploitation informatique (OS) comportant des outil(s)-support(s) de programmation (Fst, Fct) munis chacun d'une définition, on brouille ledit système d'exploitation en brouillant l'un au moins desdits outil(s)-support(s) de programmation dudit système d'exploitation par insertion dans sa définition d'au moins un paramètre de brouillage.
3. Procédé selon la revendication 2, **caractérisé en ce que** chaque outil-support (Fst, Fct) étant défini par un multiplet comportant des paramètres ou variables, on insère un paramètre de brouillage après chaque paramètre ou variable d'un multiplet.
4. Procédé selon la revendication 2, **caractérisé en ce que** chaque outil-support (Fst, Fct) étant défini par un multiplet comportant des paramètres ou variables, on insère un paramètre de brouillage avant chaque paramètre ou variable d'un multiplet.
5. Procédé selon l'une des revendications 2 à 4, **caractérisé en ce que** chaque outil-support (Fst, Fct) étant défini par un multiplet comportant des paramètres ou variables, on complète ledit brouillage en permutant au moins deux paramètres ou variables de l'un au moins desdits multiplets.
6. Procédé selon la revendication 5, **caractérisé en ce que** l'on permute tous lesdits paramètres ou toutes lesdites variables de l'un au moins desdits multiplets.
7. Procédé selon l'une des revendications 1 à 6, **caractérisé en ce que** l'on procède au brouillage en fonction d'une loi choisie.
8. Procédé selon la revendication 7, **caractérisé en ce que** ladite loi est variable.
9. Procédé selon la revendication 7, **caractérisé en ce que** ladite loi est de type pseudo-aléatoire.
10. Procédé selon la revendication 2 à 9, **caractérisé en ce que** l'on procède au brouillage de tous lesdits outil(s)-support(s) (Fst, Fct).
11. Procédé selon la revendication 2 à 10, **caractérisé en ce que** lesdits outil(s)-support(s) (Fst, Fct) sont choisis parmi des prototypes de fonction (Fct) et des fichiers inclus internes (Fst) définissant des structures.
12. Procédé selon l'une des revendications 2 à 10 en combinaison avec la revendication 11, **caractérisé en ce que** chaque prototype de fonction brouillé (Fct) est défini par un multiplet se présentant sous la forme « Fct(typei Pi, [type Dummyi,] typej Pj, [type Dummyj,]... typer Pr, [type Dummyp]) », où « typei » est le type i d'un paramètre d'appel Pi de la fonction concernée, et « Dummyi » représente au moins un paramètre de brouillage inséré, associé audit paramètre d'appel Pi.

13. Procédé selon la revendication 2 à 10 en combinaison avec l'une des revendications 11 ou 12, **caractérisé en ce que** chaque fichier inclus interne brouillé (Fst) est défini par un multiplet se présentant sous la forme Fst{typei Di; [type Dummyi;] typej Dj;[type Dummyj;]... typer Dr,[type Dummyr]}, où « typei » est le type i d'une variable Di de la structure concernée, et « Dummyi » représente au moins un paramètre de brouillage inséré, associé à ladite variable Di.

14. Dispositif informatique (D), **caractérisé en ce qu'il** comprend des moyens de traitement (MT) agencés pour mettre en oeuvre un procédé de protection de programme informatique interne (L) selon l'une des revendications 1 à 13.

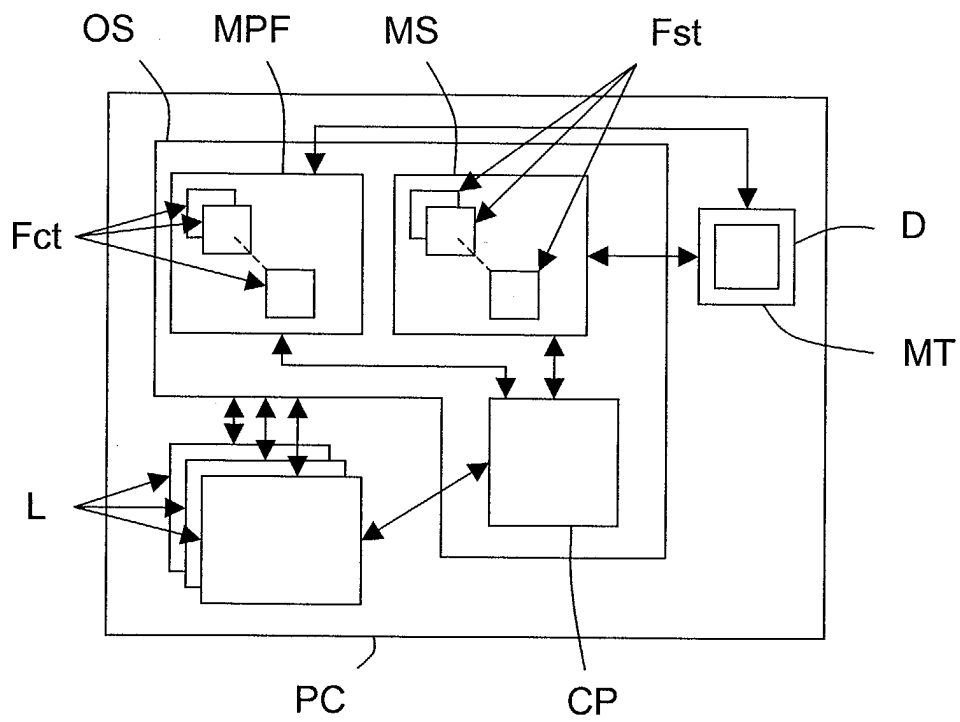


Figure unique



Office européen
des brevets

RAPPORT DE RECHERCHE EUROPEENNE

Numéro de la demande

EP 05 29 0303

DOCUMENTS CONSIDERES COMME PERTINENTS			
Catégorie	Citation du document avec indication, en cas de besoin, des parties pertinentes	Revendication concernée	CLASSEMENT DE LA DEMANDE (Int.Cl.7)
X	EP 1 376 310 A (MICROSOFT CORP) 2 janvier 2004 (2004-01-02) * page 5, alinéa 23 * * page 5, alinéa 27 - alinéa 30 * * page 8, alinéa 50 - alinéa 53 * -----	1,2, 7-11,14	G06F1/00 G06F9/00
X	US 6 668 325 B1 (COLLBERG CHRISTIAN SVEN ET AL) 23 décembre 2003 (2003-12-23) * colonne 26, ligne 21 - ligne 40 * * colonne 18, ligne 29 - ligne 39 * * colonne 5, ligne 36 - ligne 41 * * colonne 19, ligne 66 - colonne 20, ligne 6 * * colonne 2, ligne 58 - colonne 3, ligne 4 * * colonne 12, ligne 9 - ligne 18 * -----	1-14	
A	US 5 901 319 A (HIRST MICHAEL D) 4 mai 1999 (1999-05-04) * le document en entier * -----	1-14	
A	COHEN F B: "OPERATING SYSTEM PROTECTION THROUGH PROGRAM EVOLUTION" COMPUTERS & SECURITY, ELSEVIER SCIENCE PUBLISHERS. AMSTERDAM, NL, vol. 12, no. 6, 1 octobre 1993 (1993-10-01), pages 565-584, XP000415701 ISSN: 0167-4048 * alinéas [0002] - [0012] * -----	1-14	DOMAINES TECHNIQUES RECHERCHES (Int.Cl.7) G06F
Le présent rapport a été établi pour toutes les revendications			
Lieu de la recherche La Haye		Date d'achèvement de la recherche 27 avril 2005	Examineur Sigolo, A
CATEGORIE DES DOCUMENTS CITES X : particulièrement pertinent à lui seul Y : particulièrement pertinent en combinaison avec un autre document de la même catégorie A : arrière-plan technologique O : divulgation non-écrite P : document intercalaire		T : théorie ou principe à la base de l'invention E : document de brevet antérieur, mais publié à la date de dépôt ou après cette date D : cité dans la demande L : cité pour d'autres raisons & : membre de la même famille, document correspondant	

4
EPO FORM 1503 03 82 (P04C02)

**ANNEXE AU RAPPORT DE RECHERCHE EUROPEENNE
RELATIF A LA DEMANDE DE BREVET EUROPEEN NO.**

EP 05 29 0303

La présente annexe indique les membres de la famille de brevets relatifs aux documents brevets cités dans le rapport de recherche européenne visé ci-dessus.
Lesdits membres sont contenus au fichier informatique de l'Office européen des brevets à la date du
Les renseignements fournis sont donnés à titre indicatif et n'engagent pas la responsabilité de l'Office européen des brevets.

27-04-2005

Document brevet cité au rapport de recherche		Date de publication	Membre(s) de la famille de brevet(s)	Date de publication
EP 1376310	A	02-01-2004	US 2004003278 A1	01-01-2004
			EP 1376310 A2	02-01-2004
			JP 2004038966 A	05-02-2004

US 6668325	B1	23-12-2003	AU 7957998 A	25-01-1999
			CA 2293650 A1	14-01-1999
			CN 1260055 A	12-07-2000
			EP 0988591 A1	29-03-2000
			JP 2002514333 T	14-05-2002
			WO 9901815 A1	14-01-1999

US 5901319	A	04-05-1999	AUCUN	

EPO FORM P0460

Pour tout renseignement concernant cette annexe : voir Journal Officiel de l'Office européen des brevets, No.12/82